Name: _____ Section: _____

CSSE 490-01—Web Services Development

Exam 1, Apr. 3, 2017

This exam is to be solved using your computer. You will need network access to download incomplete code, install npm packages, and upload your solution. Please disable Lync, Skype for Business, IM, email, and other such communication programs before beginning the exam. Any communication with anyone other than a proctor during the exam, for any reason, *may result in a failing grade for the course.*

Allowed Resources: Open notes, open computer. Limited network access. You may use the network *only* to access your own files, the course Moodle site and web pages, the course site and web pages lined directly from it, and resources suggested by a proctor.

The instructions for the exam are included in this document.

You have a total of 2 periods to complete this exam.

Please begin by writing your name on every page of the exam. We encourage you to skim the entire exam before answering any questions.

Checkout the exam1 project from your individual remote git repository into your local repo. From a terminal (windows command prompt) window, change directory to the exam1 directory. You will need at least 3 terminal windows or tabs open as you work on the exam:

- one for the backend application,
- one for running mongod, and
- one for using the mongo shell.

From one of the terminal windows, enter the following command to run the mongo application.

```
$ mongod
```

From a second terminal window, start the mongo shell and use it to setup the database by entering these commands:

```
$ mongo
> show dbs
> use bookreviewsappdb
> db
```

The last command should display the name of the database that you will be using for the exam, which is bookreviewsappdb. You can enter any mongodb commands that you desire at this terminal. The command below will add a review to the reviews collection and give you the ability to make queries on the collection.

```
$ db.reviews.insert( "body" : "Nice read and well organized book", "book" : "9781617292422",
"createdOn" : Date.now(), "rating" : "5", "reviewer" : "me@email.com" );
```

The value for "book" is the ISBN of the text *Express in Action (Writing, building, and testing Node.js applications), by Evan M. Hahn.*

Ctrl + C exits the mongo shell.

Run the commands below in a third terminal window to start the backend application.

$ cd backend

$ npm install

$ npm start

Note: You do not need to understand all the code in this project in order to solve all the problems on the exam. You only need to understand enough to solve each problem.

This project is about a Book Review application that allows the user to display existing book reviews, add new book reviews , and update selected book reviews. Deletion of posted book reviews is not allowed in this application. Although this application only allows the user to process book reviews for one book, it can easily be extended to support multiple books. As appealing as this may seem, this is NOT the objective of this exam.

In this exam you will add the omitted functionality to the backend application in support of missing CRUD operations, allow users to register and login to the application using JSON Web Tokens (not using Rosefire or Passport), and use a Authentication: Bearer JWT header with POST and PUT requests to protected routes.

Note: TODO items have been included in the appropriate files to direct your attention to the parts of the code where you should be making changes.

It is recommended that you use the Postman - REST Client to test the API exposed by the backend application. The base address that you will use is http://localhost:8888/.

Problem	**Poss. Pts.**	**Earned**
1	18	_____
2	18	_____
3	14	_____
Total	**50**	_____

Exam 1

1. (18 points) The backend application supports GET and POST requests on the '/book-reviews/' route and GET requests on the '/book-reviews/:id' route. You might want to use Postman OR the mongo shell (see above) to add a book review to the database for the purpose of testing the exposed API. Note the collection that is being used by the application.

   a. Now add code to the backend application to provide support for DELETE requests on the '/book-reviews/:id' route. Only one file needs to be modified.

   b. Add code to the backend application to provide support for PUT requests on the '/book-reviews/:id' route. Only one file needs to be modified.

2. (18 points) The backend application is currently not secured. Any user can post and update book reviews. We would like to change that by allowing users to register and login to the application.

   a. Add code to the backend application to allow a user to register to access the protected routes of the application. A registered user should receive a JWT if the registration is successful. Usernames should be unique. Only one file needs to be modified.

   b. Add code to the backend application to provide support for user login. A logged in user should receive a JWT if the login proces is successful. Only one file needs to be modified.

3. (14 points) A user should be able to access a protected route by using a Authentication: Bearer JWT header with POST and PUT requests to said route, where JWT is replaced with the token the user received after registering or successfully logging in to the application.

Add code to secure the following requests/routes:

- POST request the /book-reviews/:id route
- PUT request on the /book-reviews/:id route

Note: only one file needs to be modified.

Be sure to commit and push your work and turn in these instructions when you leave the examination room.